Combining Insertion and Deletion in RNA-editing Preserves Regularity

E.P. de Vink*
Department of Mathematics and Computer Science, Technische Universiteit Eindhoven
Centrum Wiskunde en Informatica, Amsterdam

H. Zantema
Department of Mathematics and Computer Science, Technische Universiteit Eindhoven
Institute for Computing and Information Sciences, Radboud University Nijmegen

D. Bošnački
Department of Biomedical Engineering, Technische Universiteit Eindhoven

Abstract Inspired by RNA-editing as occurs in transcriptional processes in the living cell, we introduce an abstract notion of string adjustment, called guided rewriting. This formalism allows simultaneously inserting and deleting elements. We prove that guided rewriting preserves regularity: for every regular language its closure under guided rewriting is regular too. This contrasts an earlier abstraction of RNA-editing separating insertion and deletion for which it was proved that regularity is not preserved. The particular automaton construction here relies on an auxiliary notion of slice sequence which enables to sweep from left to right through a completed rewrite sequence.

1 Introduction

We study an elementary biologically inspired formalism of string replacement referred to as guided rewriting. Given a fixed and finite set G of strings, also called guides, a rewriting step amounts to adapting a substring towards a guide. We consider two versions of guided rewriting: *guided insertion/deletion*, which is close to an editing mechanism as encountered in the living cell, and general guided rewriting based on an *adjustment relation*, which is mathematically more amenable. For guided insertion/deletion the guide and the part of the string that is rewritten do not need to be of the same length. They are required to be equal up to occurrences of a distinguished dummy symbol. For general guided rewriting the correspondence of the guide and the substring that is rewritten is element-wise. The guide and substring are equivalent symbol-by-symbol according to a fixed equivalence relation called adjustment.

In both cases, for a finite set of guides G, only a finite set of strings can be obtained by repeatedly rewriting a given string. Starting from a language L, we may consider the extension $L_{i/d}$ of the language with all the rewrites obtained by guided insertion/deletion and the extension L_G of the language obtained by adding all the adjustment-based guided rewrites. We address the question if regularity of L implies regularity of $L_{i/d}$ and of L_G. The results of the paper state that in the case of guided insertion/deletion regularity is preserved if the strings of dummy symbols involved are bounded and that guided rewriting based on adjustment always preserves regularity.

The motivation for this work stems from transcriptional biology. RNA can be seen as strings over the alphabet $\{C,G,A,U\}$. Replication of the encoded information is one of the most essential mechanisms in

*Corresponding author, evink@win.tue.nl

life: strands of RNA are faithfully copied by the well-known processes of RNA-transcription. However, typical for eukaryotic cells, the synthesis of RNA does not yield an exact copy of part of the DNA, but a modification obtained by post-processing. The class of the underlying adjustment mechanisms is collectively called *RNA-editing*.

Abstracting away from biological details, the computational power of insertion-deletion systems for RNA-editing is studied in [14]: an insertion step is the replacement of a string uv by the string $u\alpha v$ taken from a particular finite set of triples u, α, v. Similarly, a deletion step replaces $u\alpha v$ by uv for another finite set of triples u, α, v. In [10] the restriction is considered where u and v are both empty. The approach claims full computational power, that is, they generate all recursively enumerable languages.

In the RNA-editing mechanisms occurring in nature, however, only very limited instances of these formats apply. Often only the symbol U is inserted and deleted, instead of arbitrary strings α, see e.g. [1]. Therefore, following [15], we investigate guided insertion/deletion focusing on the special role of the distinguished symbol 0, the counterpart of the RNA-base U. However, in order to prove that under this scheme regularity is preserved we extend our investigations to guided rewriting based on an abstraction adjustment relation. In fact, we prove the theorem for guided insertion/deletion by appealing to the result for guided rewriting based on adjustment.

The proof of the latter result relies on reorganizing sequences of guided rewrites into sequences of so-called slices. The point is that, since guides may overlap, each guided rewrite step adds a 'layer' on top of the previous string. In this sense guided rewriting is vertically oriented. E.g., Figure 2 in Section 5 shows six rewrite steps of the string *ebcfa* yielding the string *fbcfb* involving eight layers in total. However, in reasoning about recognition by a finite automaton a horizontal orientation is more natural. One would like to sweep from left to right, so to speak. Again referring to Figure 2, five slices can be distinguished, viz. a slice for each symbol of the string *ebcfa*. The technical machinery developed in this paper allows for a transition between the two orientations.

The organization of this paper is as follows. The biological background of RNA-editing is provided in Section 2. Section 3 presents the theorem on preservation of regularity for guided insertion-deletion. The notion of guided rewriting based on an adjustment relation is introduced in Section 4 and a corresponding theorem on preservation of regularity is presented. To pave the way for its proof, Section 5 introduces the notions of a rewrite sequence and of a slice sequence and establishes their relationship. Rewrite sequences record the subsequent guided rewrites that take place, slice sequences represent the cumulative effect of all rewrites at a particular position of the string being adjusted. In Section 6 we provide, given a finite automaton accepting a language L, the construction of an automaton for the extended language L_G with respect to a set of guides G. Section 7 wraps up with related work and concluding remarks.

Acknowledgment We acknowledge fruitful feedback from Peter van der Gulik and detailed comment from the reviewers of the MeCBIC 2012 workshop.

2 Biological motivation

This section provides a description of RNA editing from a biological perspective. In this paper we focus on the insertion and deletion of uracil in messenger RNA (mRNA) and provide abstractions of the underlying mechanism in the sequel. However, in the living cell there are different kinds of RNA editing that vary in the type of RNA that is edited and the type of editing operations. Uracil is represented by the letter U. The three other types of nucleotides for RNA, viz. adenine, guanine and cytosine are represented by the letters A, G and C, respectively.

U-insertion/deletion editing is widely studied in the mitochondrial genes of kinetoplastid protozoa [13]. Kinetoplastids are single cell organisms that include parasites like *Trypanosoma brucei* and *Crithidia fasciculata*, that can cause serious diseases in humans and/or animals. Modifications of kinetoplastid mRNA are usually made within the coding regions. These are the parts that are translated into proteins, which are the building blocks of the cells. This way coded information of the original gene can be altered and therefore expressed, i.e. translated into proteins, in a varying number of ways, depending on the environment in the cell. This provides additional flexibility as well as potential specialization of different parts of the organisms for particular functions.

Here we describe a somewhat simplified version of the mechanism for the insertion and deletion of U. More details can be found, for instance, in [13, 1, 3, 12]. For simplicity we assume that only identical letters match with one another. In reality, the matching is based on complementarity, usually assuming the so-called Crick-Watson pairs: A matches with U and G matches with C.

In general, a single step in the editing of mRNA involves two strands of RNA, a strand of messenger RNA and a strand of guide RNA, the latter typically referred to as the guide. To explain the mechanism for the insertion of uracil, let us consider an example. See Figure 1. Assume that we start of with an mRNA fragment: $u = N_1N_2N_3N_4N_5$ and the guide $g = N_2N_3UUUN_4$, where N_i can be an arbitrary nucleotide A, G or C, but not U. Obviously, there is some match between u and g involving the letters N_2, N_3, and N_4, which is partially 'spoiled' by the UUU sequence. By pairing of letters we have that g attaches to u; the matching substrings N_2N_3 and N_4 serve as anchors.

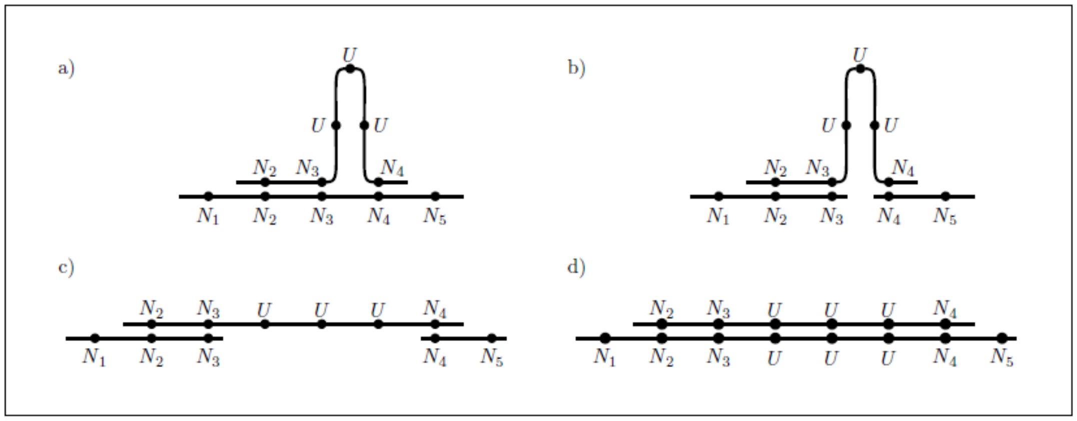

Figure 1: Various stages of guided U-insertion

By chemical reactions involving special enzymes u is split open between N_3 and N_4. The gap between the anchors is then filled by the enzyme mechanism using the guide as a template. For each letter U in the guide a U is added also in the gap. As a result the mRNA string u is transformed into $N_1N_2N_3UUUN_4N_5$.

In general, one can have more than two anchors (involving only non-U letters) in which the guide and the mRNA strand match. In that case mRNA is opened between each pair of anchors and all gaps between these anchors are filled with U such that the number of Us in the guide is matched.

A similar biochemical mechanism implements the deletion of Us from a strand of mRNA. We illustrate the deletion process on the following example. Let us assume that we have the mRNA strand $u = N_1N_2N_3UUN_4N_5$ and the guide $g = N_2N_3N_4$. Like in the insertion case, g initiates the editing by attaching itself to u at the matching positions N_2,N_3, and N_4. Only now the enzymatic complex removes the mismatching UU substring between N_3 and N_4 to ensure the perfect match between the substring and the guide. As a result the edited string $N_1N_2N_3N_4N_5$ is obtained. In general, we can have several anchoring positions on the same string. In that case, all Us between each two matching positions are removed from the mRNA.

A guide can also induce both insertions and deletions of U simultaneously. For instance the guide $N_2N_3UUUN_4$ can induce editing in parallel of the string $N_1UN_2UN_3UN_4UN_5UN_6$ which results in the string $N_1UN_2N_3UUUN_4UN_5UN_6$, where the U between N_2 and N_3 has been deleted and two U's between N_3 and N_4 have been inserted. This is done by the same biochemical mechanisms that are involved in separate insertions and deletions. Analogously as above, we can have multiple insertions and deletions induced by the same guide on the original pre-edited sequence.

The net effect of all three cases considered above is that a strand $u = xyz$, such that y equals g up to occurrences of U, is modified by the insertion and deletion mechanism and becomes a string $v = xgz$. It is noteworthy that the rewriting system that we describe in the sequel also applies to another case with the same effect. For example, consider a guide $g = N_2N_3UUUN_4$ and a pre-edited mRNA $u = N_1N_2N_3UUN_4N_5N_6$. Now, to obtain the match of the guide g and a substring y of u, a U is inserted in u, resulting in the string $v = N_1N_2N_3UUUN_4N_5N_6$. If the U subsequence in y was longer though, like in the case for $u' = N_1N_2N_3UUUN_4N_5N_6$ and $g' = N_2N_3UUN_4$, then we have that the extra U in u' is removed resulting in $v' = N_1N_2N_3UUN_4N_5N_6$.

To summarize, the mRNA editing mechanism underlying U-insertion/deletion can be interpreted as symbolic manipulations of strings. In the sequel symbol U will be denoted by 0 and obviously plays a special role. The crucial point is that in a single step some substring y is replaced by a guide g for which y and g coincide except for the symbol 0.

3 Guided insertion / deletion

Inspired by the biological scheme of editing of mRNA as discussed in the previous section, we study the more abstract notion of guided insertion and deletion and guided rewriting based on an adjustment relation in the remainder of this paper. In this section we address guided insertion and deletion, turning to guided rewriting in Section 4.

More precisely, fix an alphabet Σ_0 and distinguish $0 \notin \Sigma_0$. Put $\Sigma = \Sigma_0 \cup \{0\}$. Choose a finite set $G \subseteq \Sigma^*$, with elements g also referred to as guides. Reflecting the biological mechanism, we assume that each $g \in G$ has at least two letters and that the first and last letter of each $g \in G$ are not equal to 0. Hence, $G \subseteq \Sigma_0 {\cdot} \Sigma^* {\cdot} \Sigma_0$. Now a guided insertion/deletion step $\Rightarrow_{i/d}$ with respect to G is given by

$$u \Rightarrow_{i/d} v \iff u = xyz \wedge v = xgz \wedge g \in G \wedge \pi(y) = \pi(g)$$

where $y \in \Sigma_0 {\cdot} \Sigma^* {\cdot} \Sigma_0$, and $\pi(y)$ and $\pi(g)$ are obtained from y and g, respectively, by removing their 0s. Thus, $\pi : \Sigma^* \to \Sigma_0^*$ is the homomorphism such that $\pi(\varepsilon) = \varepsilon$, $\pi(0) = \varepsilon$ and $\pi(a) = a$ for $a \in \Sigma_0$. So, intuitively, g is anchored on the substring y of u and sequences of 0s are adjusted as prescribed by the guide g, in effect replacing the substring y by the guide g while maintaining the prefix x and suffix z.

As a simple example of a single guided insertion/deletion step, for $G = \{g\}$ with $g = bcb000ab0c$ and $u = a00bc00babcc00a00b$ we have $u \Rightarrow_{i/d} v$ for $v = a00bcb000ab0cc00a00b$. Here it holds that $u = a00 \cdot bc00babc \cdot c00a00b$, $\pi(bc00babc) = bcbabc = \pi(bcb000ab0c)$ and $v = a00 \cdot bcb000ab0c \cdot c00a00b$. Note, for the string v, being the result of a rewrite with guide g itself with only one possible anchoring, only trivial steps can be taken further. So, the operation of guided insertion/deletion with the same guide g at the same position in a string is idempotent. However, anchoring may overlap. Consider the set of guides $G = \{ aa0a, a0aa \}$, for example. Then the string aaa yields an infinite rewrite sequence

$$aaa \Rightarrow_{i/d} aa0a \Rightarrow_{i/d} a0aa \Rightarrow_{i/d} aa0a \Rightarrow_{i/d} a0aa \cdots$$

Still, from aaa only finitely many different rewrites can be obtained by insertion/deletion steps guided by this G, viz. $\{ aaa, aa0a, a0aa \}$.

The restrictions put on G exclude arbitrary deletions (possible if ε would be allowed as guide) and infinite pumping (if guides need not be delimited by symbols from Σ_0). As an illustration of the latter case, starting from the string abc and 'guide' $0ab$, the infinite sequence $abc \Rightarrow_{i/d} 0abc \Rightarrow_{i/d} 00abc \Rightarrow_{i/d} 000abc \ldots$ would be obtained. The restriction on the substring y prevents to make changes outside the scope of the guide g and forbids $a0b000c \Rightarrow_{i/d} ab0c$ by way of the guide ab.

As a first observation we show that the set $L^u_{i/d} = \{ v \in \Sigma^* \mid u \Rightarrow^*_{i/d} v \}$, for any finite set of guides G and any string u, is finite. Write $u = a_0 0^{i_1} a_1 \ldots a_{n_1} 0^{i_n} a_n$ where $a_i \in \Sigma_0$, $i_k \geqslant 0$, for some $n \geqslant 0$. In effect, a guided insertion/deletion step only modifies the substrings 0^{i_k} or leaves them as is. Therefore, after one or more guided insertion/deletion steps the substrings 0^{i_k} are strings taken from the set

$$Z^u_{i/d} = \{ 0^{i_k} \mid 1 \leqslant k \leqslant n \} \cup \{ 0^\ell \mid xa \cdot 0^\ell bz \in G,\ a,b \in \Sigma_0,\ \ell \geqslant 0 \}$$

Thus, if $u \Rightarrow^*_{i/d} v$ then $v \in \hat{L}^u_{i/d}$, where $\hat{L}^u_{i/d} = \{ a_0 z_1 a_1 \ldots a_{n_1} z_n a_n \mid z_k \in Z^u_{i/d},\ 1 \leqslant k \leqslant n \}$, i.e. $L^u_{i/d} \subseteq \hat{L}^u_{i/d}$. Since the set G is finite, it follows that $Z^u_{i/d}$ is finite, that $\hat{L}^u_{i/d}$ is finite and that $L^u_{i/d}$ is finite as well.

More generally, given a set of guides G, we define the extension by insertion/deletion $L_{i/d}$ of a language L over Σ by putting $L_{i/d} = \{ v \in \Sigma^* \mid \exists u \in L\colon u \Rightarrow^*_{i/d} v \}$. Casted to the biological setting of Section 2, L are the strands of messenger RNA, G are strands of guide RNA. Next, we consider the question whether regularity of the language L is inherited by the induced language $L_{i/d}$. Note, despite the finiteness of the insertion/deletion scheme for a single string, it is not obvious that such would hold.

For example, consider the language corresponding to the regular expression $(ab)^*$ together with the operation *sort* which maps a string w over the alphabet $\{a,b\}$ to the string $a^n b^m$ where $n = \#_a(w)$, $m = \#_b(w)$. Thus $sort(w)$ is a sorted version of w with the a's preceding the b's. Note, for $w \in (ab)^*$ there is only one string $sort(w)$, as sorting is a deterministic, hence finitary operation. However, despite $\mathscr{L}(\,(ab)^*\,)$, the language associated by the regular expression, is regular, the language

$$sort(\,(ab)^*\,) = \{ sort(w) \mid w \in (ab)^* \} = \{ a^n b^n \mid n \geqslant 0 \}$$

is *not* regular. Also, if we define the rewrite operation $ba \rightarrow_R ab$, then $\{ v \in \{a,b\}^* \mid u \rightarrow^*_R v \}$ contains shuffles of the string u, i.e. all strings over $\{a,b\}$ having the same number of a's and b's but are smaller lexicographically. Thus, the set $\{ v \in \{a,b\}^* \mid u \rightarrow^*_R v \}$ is finite for each string u. However, the language $\hat{L} = \{ v \in \{a,b\}^* \mid \exists u \in L\colon\ u \rightarrow^*_R v \}$ cannot be regular: intersection with the language of a^*b^* does not yield a regular language. More specifically, $\hat{L} \cap \mathscr{L}(a^*b^*) = \{ a^n b^n \mid n \geqslant 0 \}$. We conclude that the question of $L_{i/d}$ being regular, given regularity of the language L, is not straightforward.

With the machinery of rewrite sequences and slice sequences developed in the sequel of the paper, we will be able to prove the following for guided insertion/deletion.

Theorem 1. *In the setting above, if L is a regular language and for some number $k \geqslant 0$ it holds that no string of L or G contains k (or more) consecutive 0's, then the language $L_{i/d}$ is regular too.*

We will prove Theorem 1 by applying a more general result on guided rewriting, viz. Theorem 3 formulated in the next section and ultimately proven in Section 6. As in the notion of guided rewriting as developed in the sequel, symbols are only replaced by single symbols by which lengths of strings are always preserved, a transformation is required to be able to apply Theorem 3.

Before doing so we relate our results to those of [15]. There a relation similar to $\Rightarrow_{i/d}$ was introduced, with the only difference that in a single step either 0's are deleted or inserted, but not simultaneously. One of the conclusions of [15] is that in that setting regularity is *not* preserved, so the opposite of the main result in the present setting.

4 Guided rewriting

The idea of guided rewriting is that symbols are replaced by equivalent symbols with respect to some *adjustment relation* $\sim$. The one-one correspondence of the symbols of the string u and its guided rewrite v, enjoyed by this notion of reduction, will turn out technically convenient in the sequel.

Let Σ be a finite alphabet and $\sim$ an equivalence relation on Σ, called the *adjustment relation*. If $a \sim b$ we say that a can be adjusted to b. For a string $u \in \Sigma^*$ we write $\#u$ for its length, use $u[i]$ to denote its i-th element, $i = 1, \ldots, \#u$, and let $u[p,q]$ stand for the substring $u[p]\,u[p{+}1] \cdots u[q]$. The relation $\sim$ is lifted to Σ^* by putting

$$u \sim v \quad \textit{iff} \quad \#u = \#v \wedge \forall i = 1, \ldots, \#u \colon u[i] \sim v[i]$$

Next we define a notion of guided rewriting that involves an adjustment relation.

Definition 2. *We fix a finite subset $G \subseteq \Sigma^*$, called the set of guides.*

(a) *For $u, v \in \Sigma^*$, $g \in G$, $p \geqslant 0$, we define $u \Rightarrow_{g,p} v$, stating that v is the rewrite of u with guide g at position p, by*

$$u \Rightarrow_{g,p} v \quad \textit{iff} \quad \exists x, y, z \in \Sigma^* \colon u = xyz \;\wedge\; \#x = p \;\wedge\; y \sim g \;\wedge\; v = xgz$$

(b) *We write $u \Rightarrow v$ if $u \Rightarrow_{g,p} v$ for some $g \in G$ and $p \geqslant 0$. We use $\Rightarrow^*$ to denote the reflexive transitive closure of $\Rightarrow$. A sequence $u_1 \Rightarrow u_2 \Rightarrow \cdots \Rightarrow u_n$ is called a reduction.*

(c) *For a language L over Σ and a set of guides G we write*

$$L_G = \{\, v \in \Sigma^* \mid \exists u \in L \colon u \Rightarrow^* v \,\}$$

So, a $\Rightarrow$-step adjusts a substring to a guide in G element-wise, and L_G consists of all strings that can be obtained from a string from L by any number of such adjustments. For example, if $\Sigma = \{a,b,c\}$, $G = \{bb\}$ and $a \sim b$ but not $a \sim c$, then by a $\Rightarrow$-step two consecutive symbols not equal to c are replaced by two consecutive b's. In particular, $aaacaa \rightarrow_{bb,1} abbcaa$ and $abbcaa \rightarrow_{bb,0} bbbcaa$. We have

$$\{aaacaa\}_G = \{\, aaacaa, bbacaa, abbcaa, aaacbb, bbbcaa, abbcbb, bbacbb, bbbcbb \,\}$$

Next, we state the main result of this paper regarding guided rewriting as given by Definition 2.

Theorem 3. *Given an equivalence relation $\sim$ on Σ, let G be a finite set of guides. Suppose L is a regular language. Then L_G is regular too.*

Before going to the proof, we first show that both finiteness of G and the requirement of $\sim$ being an equivalence relation are essential. Below, for a regular expression r we write $\mathscr{L}(r)$ for its corresponding language.

To see that finiteness of G is essential for Theorem 3 to hold, let $G = \{\, ca^k cb^k c \mid k \geqslant 0 \,\}$ and $L = \mathscr{L}(ca^*ca^*c)$. Let $\sim$ satisfy $a \sim b$ but not $a \sim c$. Then all elements of L on which an adjustment is applicable are of the shape ca^kca^kc, where the result of the adjustment is ca^kcb^kc, which can not be changed by any further adjustment. So

$$L_G \cap \mathscr{L}(ca^*cb^*c) \;=\; \{\, ca^kcb^kc \mid k \geqslant 0 \,\}$$

is not regular. Since regularity is closed under intersection we conclude that L_G cannot be regular itself.

Also equivalence properties of $\sim$ are essential for Theorem 3. For $G = \{\, ab \,\}$ and $\sim \,=\, \{\, (a,b), (b,a) \,\}$ the only possible $\Rightarrow$-steps are replacing the pattern ba by ab. Note that here $\sim$ is neither reflexive nor transitive. Since ba may be replaced by ab, bubble sort on a's and b's can be mimicked by $\Rightarrow^*$, while on the other hand $\Rightarrow^*$ preserves both the number of a's and the number of b's. Hence

$$\mathscr{L}((ab)^*)_G \cap \mathscr{L}(a^*b^*) \;=\; \{\, a^kb^k \mid k \geqslant 0 \,\}$$

which proves that $\mathscr{L}((ab)^*)_G$ is not regular, again since regularity is closed under intersection.

5 Rewrite sequences and slice sequences

Fix an alphabet Σ, an adjustment relation $\sim$, and a set of guides G.

Definition 4. *A sequence $\rho = (g_k, p_k)_{k=1}^r$ of guide-position pairs is called a guided rewrite sequence for a string $u \in \Sigma^*$ if it holds that (i) $g_k \in G$, (ii) $0 \leqslant p_k \leqslant \#u - \#g_k$, and (iii) $u[p_k{+}1, p_k{+}\#g_k] \sim g_k$, for all $k = 1, \ldots, r$.*

A guide-position pair (g, p) indicates a redex for a guided rewrite with g of the string u. The position p is relative to u. For the rewrite to fit we must have $p + \#g \leqslant \#u$. The first p symbols of u, i.e. the substring $u[1,p]$, are not affected by the rewrite, as are the last $\#u - p + \#g$ symbols of u, i.e. the substring $u[p{+}\#g{+}1, \#u]$.

The sequence ρ induces a sequence of strings $(u_k)_{k=0}^r$ by putting $u_0 = u$ and u_k such that $u_{k-1} \Rightarrow_{g_k,p_k} u_k$ for $k = 1, \ldots, r$. To conclude that $u_{k-1} \Rightarrow_{g_k,p_k} u_k$ is indeed a proper guided rewrite step, in particular that we have $u_{k-1}[p_k{+}1, p_k{+}\#g_k]$, we use the assumption $u[p_k{+}1, p_k{+}\#g_k] \sim g_k$ and the fact that if $u \Rightarrow_{g,p} v$ then $u[p+1, p+\#g] \sim v[p+1, p+\#g]$. So we obtain $u \Rightarrow^* u_r$ by construction. The string u_r is referred to as the yield of ρ for u, notation *yield*(ρ). Conversely, every specific reduction from u to v gives rise to a corresponding guided rewrite sequence for u.

Definition 5. *Let $a \in \Sigma$. A sequence $s\ell = (g_i, q_i)_{i \in I}$ of guide-offset pairs, for $I \subseteq \mathbb{N}$ a finite index set, is called a slice for a and G if it holds that (i) $g_i \in G$, (ii) $1 \leqslant q_i \leqslant \#g_i$, and (iii) $a \sim g_i[q_i]$, for all $i \in I$. The slice $s\ell$ is called a slice for a string $u \in \Sigma^*$ at position n, $1 \leqslant n \leqslant \#u$, if it is a slice of $u[n]$.*

Note that in a guide-offset pair (g, q) of a slice sequence, the offset q is relative to the guide g. Since we require $1 \leqslant q \leqslant \#g$ for such a pair, the symbol $g[q]$ is well-defined. We will reserve the use of q for offsets, indices within a guide, and the use of p for positions after which a rewrite may take place, i.e. for lengths of proper prefixes of a given string.

The goal of the notion of slice is to summarize the effect of a number of guided rewrites local to a specific position within a string. The symbol generated by the last rewrite that affected the position, i.e. the particular symbol of the last element of the slice sequence, is part of the overall outcome of the total rewrite. This symbol is called the *yield* of the slice. More precisely, if $I \neq \emptyset$, the yield of a slice $s\ell$ for a symbol a is defined as $\mathit{yield}(s\ell) = g_{i_{\max}}[q_{i_{\max}}]$ where $i_{\max} = \max(I)$. In case $I = \emptyset$, we put $\mathit{yield}(s\ell) = a$. Occasionally we write $a \sim s\ell$, as for a slice $s\ell$ for a symbol a it always holds that $a \sim \mathit{yield}(s\ell)$.

A slice $s\ell$ is said to be repetition-free if $g_i = g_j \wedge q_i = q_j$ implies $i = j$. If we have $I = \emptyset$, the slice $s\ell$ is called the empty slice.

Next we consider sequences of slices, and investigate the relationship between slices on two consecutive positions in a guided rewrite sequence.

Definition 6. *A sequence $\sigma = (s\ell_n)_{n=1}^{\#u}$ is called a slice sequence for a string u if the following holds:*

- *$s\ell_n$ is a slice for u at position n, for $n = 1, \ldots, \#u$;*
- *for $n = 1, \ldots, \#u-1$, putting $s\ell_n = (g_i, q_i)_{i \in I}$ and $s\ell_{n+1} = (g'_i, q'_i)_{i \in J}$, there exists a monotone partial injection $\gamma_n : I \to J$ such that, for all $i \in I$ and $j \in J$,*
  - $i \notin \mathit{dom}(\gamma_n) \implies q_i = \#g_i$
  - $\gamma_n(i) = j \iff g_i = g'_j \wedge q_i + 1 = q'_j$
  - $j \notin \mathit{rng}(\gamma_n) \implies q'_j = 1$
- *the slices $s\ell_1$ and $s\ell_{\#u}$, say $s\ell_1 = (g_i, q_i)_{i \in I}$ and $s\ell_{\#u} = (g'_j, q'_j)_{j \in J}$, satisfy $q_i = 1$, for all $i \in I$, and $q'_j = \#g'_j$, for all $j \in J$, respectively.*

For the slices $s\ell_n$ and $s\ell_{n+1}$ the mapping $\gamma_n : I \to J$ is called the cut for $s\ell_n$ and $s\ell_{n+1}$. It witnesses that $s\ell_n$ and $s\ell_{n+1}$ match in the sense that a rewrite may end at position n, may continue for its next offset at position $n+1$, and may start at position $n+1$. Since a cut γ is an order-preserving bijection from $\mathit{dom}(\gamma)$ to $\mathit{rng}(\gamma)$, and $\mathit{dom}(\gamma)$ and $\mathit{rng}(\gamma)$ are finite, it follows that for two slices $s\ell, s\ell'$ the cut $s\ell \to s\ell'$ is unique. We write $s\ell \rightsquigarrow s\ell'$. A slice $s\ell = (g_i, q_i)_{i \in I}$ is called a start slice if $q_i = 1$ for all $i \in I$. Similarly, $s\ell$ is called an end slice if $q_i = \#g_i$ for all $i \in I$. A start slice is generally associated with the first position of the string that is rewritten, an end slice with the last position. Note, a start slice as well as an end slice are allowed to be empty. The yield of the slice sequence σ is the sequence of the yield of its slices, i.e. we define $\mathit{yield}(\sigma) = \mathit{yield}(s\ell_1) \cdots \mathit{yield}(s\ell_{\#u})$.

Example 7. *Let $\sim$ be the adjustment relation with equivalence classes $\{a,b\}, \{c,d\}, \{e,f\}$ and let the set of guides G be given by $G = \{ g_1, g_2, g_3 \}$ where $g_1 = fb$, $g_2 = ace$ and $g_3 = d$. For the string $u = ebcfa$ we consider the guided rewrite sequence $\rho = ((g_3,2), (g_1,0), (g_2,1), (g_1,0), (g_1,3), (g_1,3))$. The associated reduction looks like*

$$\mathit{ebcfa} \Rightarrow_{g_3,2} \mathit{ebdfa} \Rightarrow_{g_1,0} \mathit{fbdfa} \Rightarrow_{g_2,1} \mathit{facea} \Rightarrow_{g_1,0} \mathit{fbcea} \Rightarrow_{g_1,3} \mathit{fbcfb} \Rightarrow_{g_1,3} \mathit{fbcfb} \qquad (1)$$

Recording what happens at all of the five positions of the string u yields, for this example, the slice sequence $\sigma = (s\ell_n)_{n=1}^{5}$ given in the table at the left-hand side of Figure 2, where the slice sequence is visualized too.

For the choice of $I_1, \ldots, I_5$, the monotone partial injection γ_n, $n = 1 \ldots 4$, maps every number to itself. It is easily checked that all requirements of a slice sequence hold. The ovals covering guide-offset pairs reflect the cuts as mappings between to adjacent slices. However, they also comprise, in this situation derived from a guided rewrite sequence, complete guides. Note, $s\ell_1$ is a start slice, $s\ell_5$ is an end slice. We have for the slice sequence $\sigma = (s\ell_n)_{i=1}^{5}$ that $\mathit{yield}(\sigma) = \mathit{yield}(s\ell_1) \cdot \cdots \cdot \mathit{yield}(s\ell_5) = \mathit{fbcfb}$. Indeed, this coincides with the yield of the guided rewrite sequence ρ of (1).

	I_n	$(g_i, q_i)_{i \in I_n}$
$s\ell_1$	2,4	$2 \mapsto (g_1,1)$, $4 \mapsto (g_1,1)$
$s\ell_2$	2,3,4	$2 \mapsto (g_1,2)$, $3 \mapsto (g_2,1)$, $4 \mapsto (g_1,2)$
$s\ell_3$	1,3	$1 \mapsto (g_3,1)$, $3 \mapsto (g_2,2)$
$s\ell_4$	3,5,6	$3 \mapsto (g_2,3)$, $5 \mapsto (g_1,1)$, $6 \mapsto (g_1,1)$
$s\ell_5$	5,6	$5 \mapsto (g_1,2)$, $6 \mapsto (g_1,2)$

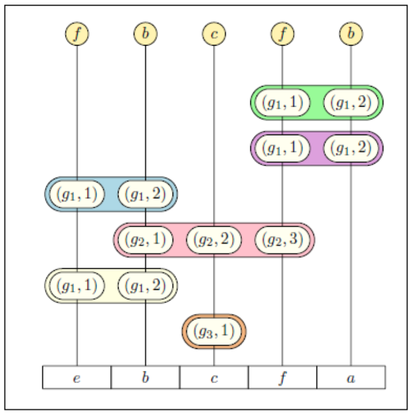

Figure 2: An example slice sequence

The rest of this section is devoted to proving that the above holds in general: Given a string and a set of guides, for every guided rewrite sequence there exists a slice sequence and for every slice sequence there exists a guided rewrite sequence. Moreover, the yield of the guided rewrite sequence and slice sequence are the same.

Theorem 8. *Let $\rho = (g_k, p_k)_{k=1}^{r}$ be a guided rewrite sequence for a string u. Then there exists a slice sequence $\sigma = (s\ell_n)_{n=1}^{\#u}$ for u such that $yield(\sigma) = yield(\rho)$.*

Proof sketch. Induction on r. If ρ is the empty rewrite sequence, we take for σ the slice sequence of n empty slices. Suppose ρ is non-empty. Let $(u_k)_{k=0}^{r}$ be the sequence of strings induced by ρ. By induction hypothesis there exists a slice sequence σ' for the first $r-1$ steps of ρ. Suppose $u_{r-1} \Rightarrow_{g_r,p_r} u_r$. The slice sequence σ is obtained by extending the slices of σ' from position $p_r{+}1$ to $p_r{+}\#g_r$ with the pairs $(g_r, n{-}p_r)$. Then,

$$\begin{aligned} yield(\sigma) &= yield(\sigma'[1,p_r]) \cdot g_r[1,\#g_r] \cdot yield(\sigma'[p_r{+}\#g_r{+}1,\#u]) \\ &= u_{r-1}[1,p_r] \cdot g_r \cdot u_{r-1}[p_r{+}\#g_r{+}1,\#u_{r-1}] \;=\; u_r \;=\; yield(\rho) \end{aligned}$$

Verification of σ being a slice sequence for u requires transitivity of $\sim$. $\square$

In order to show the reverse of Theorem 8 we proceed in a number of stages. First we need to relate individual guide-offset pairs in neighboring slices. For this purpose we introduce the ordering $\preccurlyeq$ on so-called chunks.

Definition 9. *Let $\sigma = (s\ell_n)_{n=1}^{\#u}$ be a slice sequence for u. Assume we have $s\ell_n = (\,g_{n,i}, q_{n,i}\,)_{i \in I_n}$, for $n = 1,\ldots,\#u$. Let $\gamma_n \colon I_n \to I_{n+1}$ be the cut for $s\ell_n$ and $s\ell_{n+1}$, $1 \leqslant n < \#u$. Let $\mathscr{X} = \{\,(\,g_{n,i}, q_{n,i}, i, n\,) \mid 1 \leqslant n \leqslant \#u,\ i \in I_n\,\}$ be the set of chunks of σ and define the ordering $\preccurlyeq$ on $\mathscr{X}$ by putting $(\,g,q,i,n\,) \preccurlyeq (\,g',q',i',n'\,)$ iff*

- *either $n' \geqslant n$ and there exist indexes $\ell_0, h_0, \ldots, \ell_{n'-n}, h_{n'-n}$ such that*
  - *$\ell_k, h_k \in I_{n+k}$ and $\ell_k \leqslant h_k$, $0 \leqslant k \leqslant n'-n$*
  - *$h_k \in dom(\gamma_{n+k})$ and $\gamma_{n+k}(h_k) = \ell_{k+1}$, $0 \leqslant k < n'-n$*
  - *$\ell_0 = i$ and $h_{n'-n} = i'$*
- *or $n' \leqslant n$ and there exist indexes $\ell_0, h_0, \ldots, \ell_{n-n'}, h_{n-n'}$ such that*

– $\ell_k, h_k \in I_{n'+k}$ *and* $\ell_k \leqslant h_k$, $0 \leqslant k \leqslant n-n'$
– $\ell_k \in dom(\gamma_{n'+k})$ *and* $\gamma_{n'+k}(\ell_k) = h_{k+1}$, $0 \leqslant k < n-n'$
– $h_0 = i'$ *and* $\ell_{n-n'} = i$

In the above setting with $n' \geqslant n$, we say that the sequence $\ell_0, h_0,\ \ell_1, h_1,\ \ldots,\ \ell_{n'-n}, h_{n'-n}$ is leading from $i \in I_n$ up to $i' \in I_{n'}$. Likewise for the case where $n' \leqslant n$.

For example, for the slice sequence $(s\ell_i)_{i=1}^r$ of Figure 2, to identify the guide belonging to the guide-offset pair $(g_2, 1)$ of slice $s\ell_2$, the pair is more precisely represented by the chunk $(g_2, 1, 3, 2)$, for the pair is associated with index $3 \in I_2$ of slice $s\ell_2$. Since for the cuts $\gamma_2 : I_2 \to I_3$ and $\gamma_3 : I_3 \to I_4$ we have $\gamma_2(3)$ and $\gamma_3(3) = 3$, we have $(g_2, 1, 3, 2) \preccurlyeq (g_2, 2, 3, 3) \preccurlyeq (g_2, 3, 3, 4)$ via the sequence 3,3,3,3 connects $(g_2, 1)$ and $(g_2, 2)$, and 3,3,3,3 connecting $(g_2, 2)$ and $(g_2, 3)$. (Hence the combination of sequences 3,3,3,3,3,3 connects $(g_2, 1)$ and $(g_2, 3)$ directly.) As no jumps from a low index ℓ to a high index h needs to be taken, we also have $(g_2, 1, 3, 2) \succcurlyeq (g_2, 2, 3, 3) \succcurlyeq (g_2, 3, 3, 4)$. Thus $(g_2, 1, 3, 2) \equiv (g_2, 2, 3, 3) \equiv (g_2, 3, 3, 4)\}$. In fact, $\{(g_2, 1, 3, 2), (g_2, 2, 3, 3), (g_2, 3, 3, 4)\}$ is an equivalence class for $\mathscr{X}$ corresponding to the guide g_2 (cf. Lemma 10). Differently, we have $(g_2, 1, 3, 2) \preccurlyeq (g_1, 2, 6, 5)$ relating g_2 to the fourth occurrence of g_1 via the sequence 3,3,3,3,3,5,5,5, for example. Since there is a jump here from $\ell_2 = 3$ to $h_2 = 5$, we do not have $(g_2, 1, 3, 2) \succcurlyeq (g_1, 2, 6, 5)$. This reflects that apparently the rewrite with this occurrence of g_1 is on top of part of the rewrite using g_2 as guide.

Given a slice sequence σ, the ordering $\preccurlyeq$ on the chunks of σ in $\mathscr{X}$ gives rise to a partial ordering on the set $\mathscr{X}/{\equiv}$ of equivalence classes of chunks. As we will argue, the equivalence classes correspond to guides and their ordering corresponds to the relative order in which the guides occur in a rewrite sequence ρ having the same yield as the slice sequence σ.

Lemma 10. *(a) The relation $\preccurlyeq$ on $\mathscr{X}$ is reflexive and transitive.*

(b) The relation $\equiv$ on $\mathscr{X}$ such that $x \equiv y \iff x \preccurlyeq y \wedge y \preccurlyeq x$ is an equivalence relation.

(c) The ordering $\preccurlyeq$ on $\mathscr{X}/{\equiv}$ induced by $\preccurlyeq$ on X by $[x] \preccurlyeq [y] \iff \exists x' \in [x] \exists y' \in [y] : x' \preccurlyeq y'$, makes $\mathscr{X}/{\equiv}$ a partial order. □

The next lemma describes the form of the equivalence class holding a chunk $x = (g, q, i, n)$. Using the cuts, equivalent chunks can be found backwards up to position $n{-}q{+}1$ and forward up to position $n{-}q{+}\#g$. These chunks together, $(g, 1, i_{n-q+1}, n{-}q{+}1)$, …, (g, q, i_n, n), …, $(g, \#g, i_{n-q+\#g}, n-q+\#g)$ span the guide g that is to be applied, in the rewrite sequence to be constructed.

Lemma 11. *Let $\sigma = (s\ell_n)_{n=1}^{\#u}$ be a slice sequence for a string u. Let $\mathscr{X} = \{ (g_{n,i}, q_{n,i}, i, n) \mid 1 \leqslant n \leqslant \#u,\ i \in I_n \}$ be the set of chunks and choose $x \in \mathscr{X}$, say $x = (g, q, i, n)$. Put $p = n{-}q$. Then there exist $j_1 \in I_{p+1}$, …, $j_{\#g} \in I_{p+\#g}$ such that $[x] = \{ (g, s, j_s, p+s) \mid 1 \leqslant s \leqslant \#g \}$.* □

We are now in a position to prove the reverse of Theorem 8.

Theorem 12. *Let σ be a slice sequence for a string u. Then there exists a guided rewrite sequence ρ for u such that yield(ρ) = yield(σ).*

Proof. Suppose $\sigma = (s\ell_n)_{n=1}^{\#u}$, $s\ell_n = (g_{i,n}, q_{i,n})_{i \in I_n}$, for $n = 1, \ldots, \#u$, and let $\mathscr{X} = \{ (g_{n,i}, q_{n,i}, i, n) \mid 1 \leqslant n \leqslant \#u,\ i \in I_n \}$ be the corresponding set of chunks. We proceed by induction on $\#\mathscr{X}$. Basis, $\#\mathscr{X} = 0$: In this case every slice is empty and $yield(\sigma) = yield(s\ell_1) \cdots yield(s\ell_{\#u}) = u[1] \cdot \cdots \cdot u[\#u] = u$ and the empty guided rewrite sequence for u has also yield u.

Induction step, $\#\mathscr{X} > 0$: Clearly, $\mathscr{X}/{\equiv}$ is finite and therefore we can choose, by Lemma 10, $x \in \mathscr{X}$ such that $[x]$ is maximal in $\mathscr{X}/{\equiv}$ with respect to $\preccurlyeq$. By Lemma 11 we can assume $[x] = \{ (g, s, i_s, p+s) \mid$

$1 \leqslant s \leqslant \#g \,\}$ for suitable p and indexes $i_s \in I_{p+s}$, for $s = 1,\dots,\#g$. Note, by maximality of $[x]$, the indexes i_s must be the maximum of I_{p+s}. In particular, $\mathit{yield}(\,\sigma\,)[p+s] = \mathit{yield}(\,s\ell_{p+s}\,) = g[s]$, for $s = 1,\dots,\#g$.

Now, consider the slice sequence $\sigma' = (\,s\ell'_n\,)_{n=1}^{\#u}$ where

$$s\ell'_n = \begin{cases} s\ell_n & \text{for } n = 1,\dots,p \text{ and } n = p{+}\#g{+}1,\dots,\#u \\ (\,g_{i,n}, q_{i,n}\,)_{i \in I_n \setminus \{i_{n-p}\}} & \text{for } n = p{+}1,\dots,p{+}\#g \end{cases}$$

So, the slice sequence σ' is obtained from the slice sequence σ by leaving out the guide-offset pairs related to the particular occurrence of g.

Let $\mathscr{X}'$ be the set of chunks of σ'. Then $\#\mathscr{X}' < \#\mathscr{X}$. By induction hypothesis we can find a guided rewrite sequence $\rho' = (\,g'_k, p'_k\,)_{k=1}^{r}$ for u such that $\mathit{yield}(\,\rho'\,) = \mathit{yield}(\,\sigma'\,)$. Define the guided rewrite sequence $\rho = (\,g_k, p_k\,)_{k=1}^{r+1}$ by $g_k = g'_k$, $p_k = p'_k$ for $k = 1,\dots,r$ and $g_{r+1} = g$, $p_{r+1} = p$. We have $0 \leqslant p \leqslant \#u{-}\#g$ and $u[p{+}1, p{+}\#g] \sim g$ since $s\ell_{p+1},\dots,s\ell_{p+\#g}$ are slices for $u[p{+}1],\dots,u[p{+}\#g]$, respectively. So, ρ is a well-defined guided rewrite sequence for u.

It holds that $\mathit{yield}(\,\rho'\,) \Rightarrow_{g,p} \mathit{yield}(\,\rho\,)$ as ρ extends ρ' with the pair (g,p). Therefore,

$$\mathit{yield}(\rho)[n] = \begin{cases} \mathit{yield}(\rho')[n] & \text{for } n = 1,\dots,p \text{ and } n = p{+}\#g{+}1,\dots,p{+}\#g \\ g[n{-}p] & \text{for } n = p{+}1,\dots,p{+}\#g \end{cases}$$

From this it follows, for any index n, $1 \leqslant n \leqslant p$ or $p{+}\#g{+}1 \leqslant n \leqslant \#u$, that $\mathit{yield}(\,\rho\,)[n] = \mathit{yield}(\,\rho'\,)[n] = \mathit{yield}(\,\sigma'\,)[n] = \mathit{yield}(\,\sigma\,)[n]$, and for any index n, $p{+}1 \leqslant n \leqslant p{+}\#g$, that $\mathit{yield}(\,\rho\,)[n] = g[n{-}p] = \mathit{yield}(\,\sigma\,)[n]$. As $\#\mathit{yield}(\,\rho\,) = \#\mathit{yield}(\,\sigma\,) = \#u$, we obtain $\mathit{yield}(\,\rho\,) = \mathit{yield}(\,\sigma\,)$, as was to be shown. □

For the slice sequence $(s\ell_i)_{i=1}^{5}$ of Figure 2 we have the following equivalence classes of chunks:

$$\begin{array}{lcllcl} G_3 & = & \{\,(g_3,1,1,3)\,\} & G_2 & = & \{\,(g_2,1,3,2), (g_2,2,3,3), (g_2,3,3,4)\,\} \\ G_1^1 & = & \{\,(g_1,1,2,1), (g_1,2,2,2)\,\} & G_1^3 & = & \{\,(g_1,1,5,4), (g_1,2,5,5)\,\} \\ G_1^2 & = & \{\,(g_1,1,4,1), (g_1,2,4,2)\,\} & G_1^4 & = & \{\,(g_1,1,6,4), (g_1,2,6,5)\,\} \end{array}$$

Moreover, $G_3 \preccurlyeq G_1^1 \preccurlyeq G_2$, $G_2 \preccurlyeq G_1^2$ and $G_2 \preccurlyeq G_1^3 \preccurlyeq G_1^4$. A possible linearization is $G_3 \preccurlyeq G_1^1 \preccurlyeq G_2 \preccurlyeq G_1^3 \preccurlyeq G_1^4 \preccurlyeq G_1^2$. This corresponds to the rewrite sequence

$$\mathit{ebcfa} \Rightarrow_{g_3,2} \mathit{ebdfa} \Rightarrow_{g_1,0} \mathit{fbdfa} \Rightarrow_{g_2,1} \mathit{facea} \Rightarrow_{g_1,3} \mathit{facfb} \Rightarrow_{g_1,3} \mathit{facfb} \Rightarrow_{g_1,0} \mathit{fbcfb}$$

Note that the yield *fbcfb* of this rewrite sequence is the same as the yield of the sequence (1) of Example 7. However, here the second rewrite with g_1 of (1) has been moved to the end. This does not effect the end result as the particular rewrites do not overlap.

6 Guided rewriting preserves regularity

Given a language L and a set of guides G, the language L_G is given as the set $\{\, v \in \Sigma^* \mid \exists u \in L\colon\, u \Rightarrow^* v \,\}$. One of the main results of this paper, Theorem 3 formulated in Section 4, states that if L is regular than L_G is regular too. We will prove the theorem by constructing a non-deterministic finite automaton accepting L_G from a deterministic finite automaton accepting L. The proof exploits the correspondence of rewrite sequences and slice sequences, Theorem 8 and Theorem 12. First we need an auxiliary result to assure finiteness of the automaton for L_G.

Lemma 13. *Let G be a finite set of guides. Let $Z = \{\, s\ell \mid s\ell$ repetition-free slice for a and G, $a \in \Sigma \,\}$. Then Z is finite. Moreover, for every string u and every rewrite sequence ρ for u, there exists a slice sequence σ for u consisting of slices from Z only such that $yield(\sigma) = yield(\rho)$.*

Proof sketch. Finiteness of Z is immediate: there are finitely many guide-offset pairs (g,q), hence finitely many repetition-free finite sequences of them. Thus, there are only finitely many repetition-free slices.

Now, let ρ be a rewrite sequence for a string u. By Theorem 8 we can choose a slice sequence σ' such that $yield(\sigma') = yield(\rho)$. Suppose $\sigma' = (s\ell_n)_{n=1}^{\#u}$ and $s\ell_n = (g_{i,n}, q_{i,n})_{i \in I_n}$ for $n = 1,\dots,\#u$. By Lemma 11 it follows that given a repeated guide-offset pair (g,q), say $(g,q) = (g_{i,n}, q_{i,n})$ and $(g,q) = (g_{j,n}, q_{j,n})$ for indexes $i < j$ in I_n, we can delete the complete equivalence class of (g_i, q_i, i, n) from slices $s\ell_{n-q+1}$ to $s\ell_{n-q+\#g}$, while retaining a slice sequence. In fact, we are removing the 'lower' occurrence of the guide g. Moreover, the resulting slice sequence has the same yield as for all slices the topmost guide-offset pair remains untouched. The existence of a repetition-free slice sequence σ such that $yield(\sigma) = yield(\sigma')$, hence $yield(\sigma) = yield(\rho)$, then follows by induction on the number of repetitions. □

As a corollary we obtain that every rewrite sequence has a repetition-free equivalent, an intuitive result which requires some technicalities though to obtain directly.

We are now prepared to prove that guided rewriting preserves regularity.

Proof of Theorem 3. Without loss of generality $\varepsilon \notin L$. Let $M = (\Sigma, Q, \rightarrow, q_o, F)$ be a DFA accepting L. We define the NFA $M' = (\Sigma, Q', \rightarrow', q_0, F')$ as follows: Let q_F be a fresh state. Put $Q' = Q \cup (Q \times Z) \cup \{q_F\}$ with Z as given by Lemma 13, $F' = \{q_F\}$ and

$$
\begin{array}{rl}
q_0 \xrightarrow{\varepsilon}' q_0 \times \zeta & \text{if } \zeta \text{ is a start slice} \\
q \times \zeta \xrightarrow{b}' q' \times \zeta' & \text{if } q \xrightarrow{a} q',\ a \sim \zeta,\ yield(\zeta) = b,\ \zeta \rightsquigarrow \zeta' \\
q \times \zeta \xrightarrow{b}' q_F & \text{if } \exists q' \colon q \xrightarrow{a} q' \in F,\ a \sim \zeta,\ yield(\zeta) = b,\ \zeta \text{ is an end slice}
\end{array}
$$

Note, by Lemma 13, Q' is a finite set of states. The automaton M' has only one final state, viz. q_F.

Suppose $v \in L_G$. Then there exist $u = a_1 \cdots a_s \in L$, a rewrite sequence $\rho = (g_k, p_k)_{k=1}^{r}$ and strings $u_0, u_1, \dots, u_r$ such that $u = u_0$, $u_{k-1} \Rightarrow_{g_k, p_k} u_k$ for $k = 1,\dots,r$, and $v = u_r$. Let, by Theorem 8 and Lemma 13, σ be a slice sequence for u of repetition-free slices with $yield(\sigma) = yield(\rho)$. Say $\sigma = (s\ell_n)_{n=1}^{\#u}$ and $s\ell_n = (g_{i,n}, q_{i,n})_{i \in I_n}$ for $n = 1,\dots,\#u$. Let $q_0 \xrightarrow{a_1} q_1 \cdots \xrightarrow{a_s} q_s \in F$ be an accepting computation of M for u. Then $q_0 \xrightarrow{\varepsilon}' q_0 \times s\ell_1 \xrightarrow{b_1}' \cdots q_{s-1} \times s\ell_s \xrightarrow{b_s}' q_F$ is an accepting computation of M'. Since we have $b_1 \cdots b_s = yield(s\ell_1) \cdots yield(s\ell_s) = yield(\sigma) = v$, it follows that $v \in \mathscr{L}(M')$. So, $L_G \subseteq \mathscr{L}(M')$.

Let $v = b_1 \cdots b_s$ be a string in $\mathscr{L}(M')$. Given the definition of the transition relation on M', we can find states $q_0, q_1, \dots, q_{s-1}$, repetition-free slices $s\ell_1, \dots s\ell_s$ such that $s\ell_n \rightsquigarrow s\ell_{n+1}$ for $n = 1,\dots,s-1$, and a computation $q_0 \xrightarrow{\varepsilon}' q_0 \times s\ell_1 \xrightarrow{b_1}' \cdots q_{s-1} \times s\ell_s \xrightarrow{b_s}' q_F$. Thus, there exist a final state q_s and a computation $q_0 \xrightarrow{a_1} q_1 \cdots q_{s-1} \xrightarrow{a_s} q_s \in F$ such that $a_n \sim s\ell_s$ for $n = 1,\dots,s$, i.e. $s\ell_n$ is a slice for a_n. Put $u = a_1 \cdots a_s$. Then $u \in L$, $(s\ell_n)_{n=1}^{\#u}$ is a slice sequence for u and $yield(\sigma) = v$. By Theorem 12 we can find a rewrite sequence ρ for u such that $yield(\rho) = yield(\sigma) = v$. It follows that $u \Rightarrow^* v$ and $v \in L_G$. Thus, $\mathscr{L}(M') \subseteq L_G$. We conclude that $L_G = \mathscr{L}(M')$ and regularity of L_G follows. □

Since $L \subseteq L_G$ the automaton M' should accept any word $a_1 \dots a_s \in L$, $s > 0$. This can be verified as follows. Let ζ_i be the empty slice for a_i, $i = 1 \dots s$. Then $a_i \sim \zeta_i$, i.e. $a_i = yield(\zeta_i)$, which holds by definition. Moreover, ζ_1 is a start slice, $\zeta_i \rightsquigarrow \zeta_{i+1}$ for $i = 1 \dots s-1$, and ζ_s is an end slice. It follows that we can turn an accepting computation of M, say $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \cdots \xrightarrow{a_s} q_s \in F$ into an accepting computation of M': $q_0 \xrightarrow{\varepsilon}' q_0 \times \zeta_1 \xrightarrow{a_1}' q_1 \times \zeta_2 \xrightarrow{a_2}' \cdots \xrightarrow{a_{s-1}}' q_{s-1} \times \zeta_s \xrightarrow{a_s}' q_F$.

We now return to a proof of Theorem 1 formulated in Section 3 for which we want to apply Theorem 3. For the latter theorem to apply we need a preparatory transformation. The point is, in the setting of guided insertion/deletion, strings are allowed to grow or shrink while guided insertions and deletions are being applied, whereas in the setting of guided rewriting the strings do not change length.

The key idea of the transformation is that every group of 0's is compressed to a single symbol. Let a language L over Σ and a number k be given by Theorem 1. So, L does not contain strings with k or more 0s. We introduce k fresh symbols $0_0, 0_1, \ldots, 0_{k-1}$. Put $\Theta = \{\, 0_0, 0_1, \ldots, 0_{k-1} \,\}$. For any string u over Σ not containing the substring 0^k, i.e. not containing k or more zeros, we define the string $\bar{u}$ over the alphabet $\overline{\Sigma} = (\Sigma \setminus \{0\}) \cup \{0_0, 0_1, \ldots, 0_{k-1}\}$ that is obtained from u by replacing every maximal pattern 0^i by the single symbol 0_i. Note, between two consecutive non-zero letters ab the symbol 0_0 is interspersed. For instance, for $k \geqslant 3$, $\overline{10023} = 1 0_2 2 0_0 3$. Also note, that the compression scheme constitutes a 1–1 correspondence of $\Sigma^* \cap \{\, w \mid w \text{ has no substring } 0^k \,\}$ and $\big(\Theta \cdot \Sigma_0\big)^* \cdot \Theta$.

Next, we show that the above operation of compressing groups of 0s preserves regularity using basic closure properties of the class of regular languages, cf. [7, Section 3].

Lemma 14. *Let L be a language without strings containing 0^k and let $\overline{L} = \{\bar{u} \mid u \in L\}$. Then $\overline{L}$ is regular if and only if L is regular.*

Proof. The language L is the homomorphic image of $\overline{L}$ for $h \colon \overline{\Sigma}^* \to \Sigma^*$ with $h(0_i) = 0^i$ and $h(a) = a$ otherwise. So, if $\overline{L}$ is regular, so is L. Reversely, $\overline{L} = (\Theta \cdot \Sigma)^* \cdot \Theta \cap h^{-1}(L)$. Hence, if L is regular, so is $\overline{L}$. □

With the above lemma in place we can give a proof of the preservation of regularity by guided insertion/deletion.

Proof of Theorem 1. Let k be as given by the statement of the theorem. Obtain $\overline{L}$ by applying the compression of strings 0^i, for $i < k$, changing from the alphabet Σ to $\overline{\Sigma}$, as introduced above. By Lemma 14 we then have that $\overline{L}$ is regular. Let $\overline{G}$ be obtained from G, again by compression of strings 0^i, for $i < k$. Then $\overline{G}$ is a finite set of guides with respect to $\overline{\Sigma}$. Now let the adjustment relation $\sim$ be the equivalence relation on $\overline{\Sigma}$ generated by $0_i \sim 0_j$, $0 \leqslant i, j < k$. By Theorem 3 we obtain that $\overline{L}_{\overline{G}}$ is regular.

Next we note that if $u \Rightarrow_{i/d} v$ with respect to Σ, then $\bar{u} \Rightarrow \bar{v}$ with respect to $\overline{\Sigma}$. Vice versa, if $\bar{u} \Rightarrow \bar{v}$ and there exist (unique) u and v such that u, v map to $\bar{u}, \bar{v}$ under compression, then $u \Rightarrow_{i/d} v$. It follows that $\overline{L}_{\overline{G}}$ and $\overline{L_{i/d}}$ coincide. Finally, by another application of Lemma 14, we conclude that $L_{i/d}$ is regular. □

7 Related work and concluding remarks

In this paper we have discussed abstract concepts of guided rewriting: a more flexible notion focusing on insertions and deletions of a dummy symbol, another more strict notion based on an equivalence relation. Given a language L we considered the extended languages $L_{i/d}$ and L_G comprising the closure of L for the two types of guided rewriting with guides from a finite set G. In particular, as our main result we proved that these closures preserve regularity. For doing so we investigated the local effect of guided rewriting on two consecutive string positions, leading to a novel notion of a slice sequence. Finally, the theorem for adjustment-based rewriting was proved by an automaton construction exploiting a slice sequence characterization of guided rewriting. Via a compression scheme for strings of dummy symbols, the theorem for guided insertion/deletion followed.

Preservation of regularity by closing a language with respect to a given notion of rewriting arises as a natural question. In Section 3 we observed that by closing the regular language $\mathscr{L}(\,(ab)^*\,)$ under rewriting with respect to the single rewrite rule $ba \to ab$ the resulting language is not regular. So, by arbitrary string rewriting regularity is not necessarily preserved. A couple of specific rewrite formats have been proposed in the literature. In [6] it was proved that regularity is preserved by deleting string rewriting, where a string rewriting system is called deleting if there exists a partial ordering on its alphabet such that each letter in the right-hand side of a rule is less than some letter in the corresponding left-hand side. In [9] it was proved that regularity is preserved by so-called period expanding or period reducing string rewriting. When translated to the setting of [15], as also touched upon in Section 3, our present notion of guided insertions and deletions allows for simultaneous insertion and deletion of the dummy symbol. A phenomenon also supported by biological findings. Remarkably, the more liberal guided insertion/deletion approach preserves regularity, whereas in the more restricted mechanism of [15], not mixing insertions and deletions per rewrite step, regularity is not preserved. As another striking difference with the mechanism of [15], for that format it was shown that strings u,v of length n exist satisfying $u \Rightarrow^* v$, but the length of the reduction is at least exponential in n. In our present format this is not the case: we expect that our slice characterization of guided rewriting serves to prove, that if $u \Rightarrow^* v$ then there is always a corresponding reduction of length linear in the length of u and v. Details have not been worked out yet.

The notion of splicing, inspired by DNA recombination, has been proposed by Head in [5]. A so-called splicing rule is a tuple $r = (u_1, v_1; u_2, v_2)$. Given two words $w_1 = x_1u_1v_1y_1$ and $w_2 = x_2u_2v_2y_2$ the rule r produces the word $w = x_1u_1v_2y_2$. So, the word w_1 is split in between u_1 and v_1, the word w_2 in between u_2 and v_2 and the resulting subwords x_1u_1 and v_2y_2 are recombined into the word w. For splicing a closure result, reminiscent to the one for guided rewriting considered in this paper, has been established. Casted in our terminology, if L is a regular language and S is a finite set of splicing rules, then L_S is regular too, cf. [8, 11]. Here, L_S is the least language containing L and closed under the splicing rules of S.

The computational power of a variant of insertion-deletion systems was studied in [14]. There deletion means that a string $u\alpha v$ is replaced by uv for a predefined finite set of triples u,α,v, while by insertion a string uv is replaced by $u\alpha v$ for another predefined finite set of triples u,α,v. This notion of insertion-deletion is quite different from ours, and seems less related to biological RNA editing. In the same vein are the guided insertion/deletion systems of [2]. There a hierarchy of classes of insertion/deletion systems and related closure properties are studied. Additionally, a non-mixing insertion/deletion system that models part of the RNA-editing for kinetoplastids is given. A rather different application of term rewriting in the setting of RNA is reported in [4], where the rewrite engine of Maude is exploited to predict the occurrence of specific patterns in the spatial formation of RNA, with competitive precision compared to techniques that are more frequently used in bioinformatics.

Possible future work includes investigation of preservation of context-freedom and of lifting the bound on the number of consecutive 0's in Theorem 1. More specifically, for a context-free language L, does it hold, for a finite set of guides G, that L_G is context-free too? Considering the set of guides, a generalization to regular sets G is worthwhile studying. Note that the counter-example given in Section 4 involves a non-regular set of guides. So, if L is regular and G is regular, do we have that L_G is regular? Similarly for L context-free. We also plan to consider guided rewriting based on other types of adjustment relations. In particular, rather than comparing strings symbol-by-symbol, one can consider two strings compatible if they map to the same string for a chosen string homomorphism. A prime example would be the erasing of the dummy 0 in the context of Section 3 for which we conjecture a variant of Theorem 3 to hold.

References

[1] J.D. Alfonzo, O. Thiemann & L. Simpson (1997): *The Mechanism of Insertion/Deletion RNA Editing in Kinetoplastid Mitochondria*. Nucleic Acids Research 25(19), pp. 3751–3759, doi:10.1093/nar/25.19.3571.

[2] F. Biegler, M.J. Burrell & M. Daley (2007): *Regulated RNA Rewriting: Modelling RNA Editing with Guided Insertion*. Theoretical Computer Science 387(2), pp. 103–112, doi:10.1016/j.tcs.2007.07.030.

[3] B. Blum, N. Bakalara & L. Simpson (1990): *A Model for RNA Editing in Kinetoplastid Mitochondria: RNA Molecules Transcribed From Maxicircle DNA Provide the Edited Information*. Cell 60, pp. 189–198, doi:10.1016/0092-8674(90)90735-W.

[4] Xuezheng Fu, Hao Wang, W. Harrison & R. Harrison (2005): *RNA Pseudoknot Prediction using Term Rewriting*. In: Proc. BIBE'05, Minneapolis, IEEE Computer Society, pp. 169–176, doi:10.1109/BIBE.2005.50.

[5] T. Head (1987): *Formal Language Theory and DNA: An Analysis of the Generative Capacity of Specific Recombinant Behaviors*. Bulletin of Mathematical Biology 49(6), pp. 737–759, doi:10.1016/S0092-8240(87)90018-8.

[6] D. Hofbauer & J. Waldmann (2004): *Deleting String Rewriting Systems Preserve Regularity*. Theoretical Computer Science 327, pp. 301–317, doi:10.1016/j.tcs.2004.04.009.

[7] J.E. Hopcroft & J.D. Ullman (1979): *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.

[8] K. Cullik II & T. Harju (1991): *Splicing Semigroups and Dominoes and DNA*. Discrete Applied Mathematics 31(3), pp. 261–271, doi:10.1016/0166-218X(91)90054-Z.

[9] P. Leupold (2008): *On Regularity-Preservation by String-Rewriting Systems*. In C. Martín-Vide, F. Otto & H. Fernau, editors: Proc. LATA 2008, LNCS 5196, pp. 345–356, doi:10.1007/978-3-540-88282-4_32.

[10] M. Margenstern, G. Paun, Y. Rogozhin & S. Verlan (2005): *Context-free Insertion-deletion Systems*. Theoretical Computer Science 330, pp. 339–348, doi:10.1016/j.tcs.2004.06.031.

[11] D. Pixton (1996): *Regularity of Splicing Languages*. Discrete Applied Mathematics 70(1), pp. 57–79, doi:10.1016/0166-218X(95)00079-7.

[12] H. van der Spek, G.J. Arts, R.R. Zwaal, J. van den Burg, P. Sloof & R. Benne (1991): *Conserved Genes Encode Guide RNAs in Mitochondria of* Crithidia Fasciculata. The EMBO Journal 10(5), pp. 1217–1224.

[13] K. Stuart, T.E. Allen, S. Heidmann & S.D. Seiwert (1997): *RNA editing in kinteoplastid protozoa*. Micorbiology and Molecular Biology Reviews 61(1), pp. 105–120.

[14] A. Takahara & T. Yokomori (2003): *On the Computational Power of Insertion-Deletion Systems*. Natural Computing 2(4), pp. 321–336, doi:10.1023/B:NACO.0000006769.27984.23.

[15] H. Zantema (2010): *Complexity of Guided Insertion-Deletion in RNA-Editing*. In A.-H. Dediu, H. Fernau & C. Martín-Vide, editors: Proc. LATA 2010, LNCS 6031, pp. 608–619, doi:10.1007/978-3-642-13089-2_51.